Natalia Ratnikova, Andrea Sciaba, Stephan Wynhoff, CMS

DISTRIBUTING APPLICATIONS IN DISTRIBUTED COMPUTING ENVIRONMENT

Software distribution is a process of delivering software products to the users. The complexity of this task increases in the highly geographically dispersed collaborations, such as modern HEP experiments with a multilevel-level hierarchy of the Regional Centers.

Frequent software releases, typical for the intensive software development phase, add extra complexity to the software distribution task.

Two plots on the right side of the poster show the CMS software release history at CERN for the last six month, and the versions installed at Fermilab on request of local physicists for the same period.

The distribution system not only addresses the question, how to make software available to the users in a reasonable time, but also how to insure that software that is used in different research centers produces the same result? What are the implications on the software distribution in the GRID computing environment?

The release management is performed at CERN, a tier-0 center. CMS is using Software SCRAM (Configuration Release And Management) tool to set up, build, and install software releases. The scram built-in software distribution mechanism allows to download remotely the project sources and list of required external products from the cvs repository.

The local installation turned out to be quite cumbersome, primarily due to a big number of external packages. That caused certain problems for the smaller regional centers, that did not have required expertise in the configuration and build tools and understanding of the projects inter-dependencies.

Fermilab, as a tier-1 center in the US, provided software installation on the distributed file system (AFS) for the FNAL users and remote sites with good connectivity, and packaging CMS software on the hard disks for the remote sites with unstable or slow network connection. In parallel we started working on the automated tool for the binary distribution.

Thus at Fermilab was developed DAR (Distribution After Release) tool for packing and installation of the application-base distributions. DAR creates a self-contained image of the application environment. Originally DAR was developed as an extension to the SCRAM tool. It is using the information about the run time environment of the CMS software applications, but no detailed information about the internal structure of the project.
Therefore practically the same approach can be used for other non-scram managed products.

DAR is robust, fully automated and extremely simple in use: one command to create distribution, and one command to install a distribution. It can be easily used directly or integrated into any complicated and intelligent system. The installation is self-contained, it does not rely on any external stuff except for compatible operating system and few system libraries. There is no external dependencies, and installation resides under one directory.
No special bookkeeping is required. To preserve existing installations, DAR does not allow to install distribution in the same location, once it has been already installed. The whole installation can be removed in one command.

To create distribution user needs to login to the node where he can run application in the standard working environment (e.g. at CERN) and execute the command:

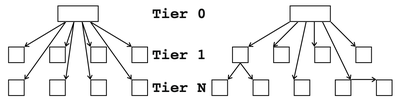
```
dar -c $CMS_PATH/Releases/ORCA/ORCA_6_1_1
$SCRATCH
```

DAR will analyse the run time environment, create the distribution tar file and print out the installation instructions. To install use:

```
dar -i ORCA_6_1_1_dar.tar.gz $MY_LOCAL_INSTALLATION
```

DAR will unpack distribution, create local installation in specified location and tune the environment scripts according to the local installation path. Finally, it will advise, how to set the run time environment (both csh-like and sh-like environment are supported).

There is no restriction to go to CERN node. DAR can create distribution from any trusted installation, i.e. DAR works equally well with centralized or decentralized distribution model:



Short development history track:

* First DAR prototype: end of summer 2001.
* Presented concept and algorithm on the CHEP in September
* First public release in November
* Validated and adopted by CMS production and used for the Big Min Bias production : end 2001- beginning 2002.
* New version to provide some improvements recommended by the production team.
* Provided software installation for the CMS SPRING02 world-wide production, running on 21 clusters in 11 Regional centers, and successfully completed in June 2002.

Abstraction of job execution from the specifics of build and configuration systems makes DAR distributions attractive for the GRID applications. Presently they are used on several GRID test beds. Since DAR created applications are self-contained, they can e.g. migrate to the data and be used for the data processing or analysis. They can be considered as analogue of the statically linked executables for the case of shared libraries, plus required run time environment.

Basically DAR can be used to run demonstration, batch jobs, check the binary compatibility and many other cases, when there is no need to rebuild the application.

However DAR does not replace the standard project installation, which is used for the software development. For this purposes more appropriate is to use the rpm based distribution, which is described below. Rpms simplify the installation procedure compare to source distribution, and provides full flexibility and functionality of the CMS software system.

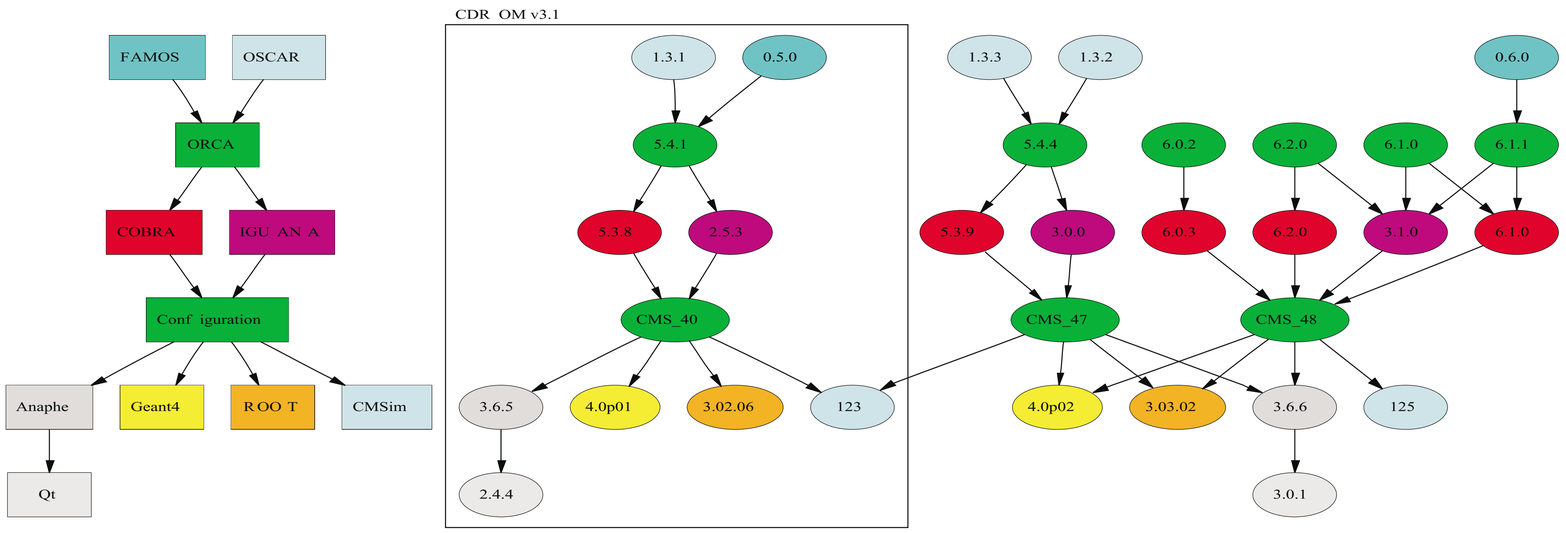
The rpm files are created by Andrea Sciaba and Stephan Wynhoff. They are distributed via network or on the CDs with a set of installation scripts, that create convenient initial CMS environment and proper standard installation. For the case of customized installations rpms can be used directly by experts.

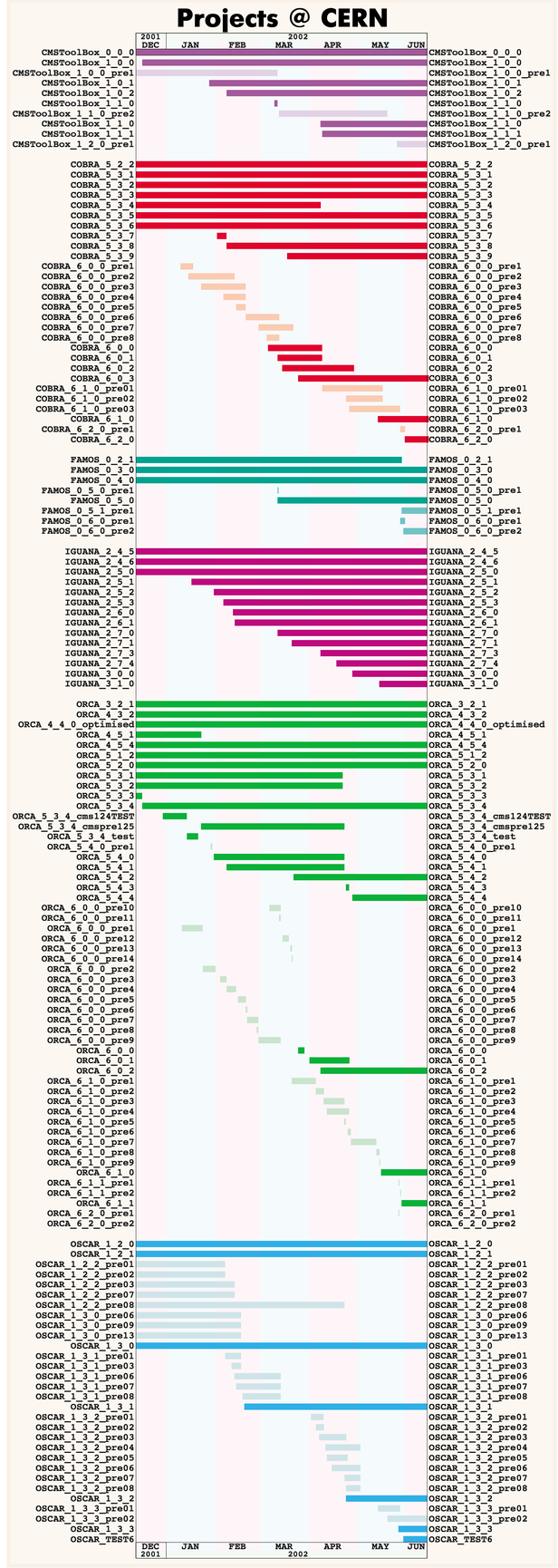
In conclusion, a few general remarks on the software distribution. Most common requirements to distributions are :

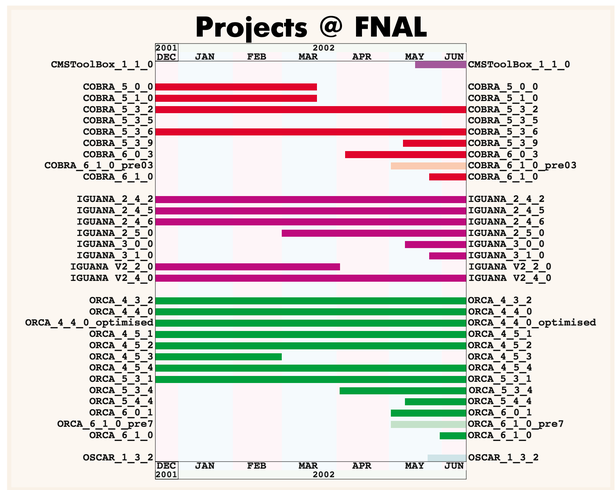
- to be relocatable
- high level of automation
- minimized external dependences
- friendly interface
- no root access required

In spite of various approaches, all models have basically the same steps:

1) packaging	2) transfer	3) unpack and install
Most distribution systems tend to keep distributions in one file. Splitting into several "portions" sometimes is reasonable, if one part of the kit can be used independently (or skipped). This makes installation more flexible, but puts more burden on the tracking dependencies systems.	download from WEB, ftp, scp or hard copy (CD).	This may involve the disk space maintenance and bookkeeping.







Distribution with rpm files

RPM is an open software packaging system developed by Red Hat, commonly used by several Linux distributions and ported also to other UNIX platforms (http://www.rpm.org/). The main advantages of distributing software by means of RPM compared with other methods (as plain tar files) can be summarized as follows:

- the ability of performing configuration tasks automatically
- the knowledge of the dependencies among different packages
- the simplicity of use

It is easy to recognize why these features are particularly suited to the CMS software environment, as most of the CMS software requires to be properly configured, after being installed, and is linked to several other pieces of software (CMS specific or not) by an intricate network of dependencies. Also the necessity of deploying the software on large computing farms makes obvious the advantage of using RPM as an installation tool.
The CMS rpm distribution was born primarily to meet the requirement of the European DataGrid project of using RPM to package all the software to be installed on its testbeds.
Specifically, DataGrid adopts a tool called LCFG (http://www.lcfg.org/) for the installation, configuration and management of its dedicated testbeds, which is completely RPM-based. To meet a requirement of some testbed sites, with some restrictions on the directory paths available to install additional software, the CMS rpm files were made to be completely "relocatable", i.e. they can be properly installed in an arbitrary directory.
Subsequently, the CMS rpm distribution has been adopted by the CMS collaboration as the recommended distribution for complete installations of the CMS software.
Briefly, an rpm file is built from a "spec" file, which contains a preamble with the description of the package and a few shell scriplets which perform the compilation and the installation. One of more tarballs, specified in the spec file, contain the source files of the program.
For the CMS software, the spec files are generated with a set of scripts which take as arguments the version number of the program to be packaged and the default installation path (which can be changed at installation time). The tarballs do not contain just the source files, but also the binaries compiled on a reference Linux machine. Pre- and post-install and uninstall scripts in the spec file are used to perform any checks and operations necessary to configure the software or to remove it from the system.
Presently, the rpm build is not automated, but performed manually whenever a new version of a package is released and a tarball is prepared. However, the rpm build is accomplished very easily and in a matter of minutes. The whole compilation and build procedure is planned to be automated in the near future, though.
The rpm installation can be done in different ways, for example manually, with a unique shell script, or with some cluster installation and managing system (the DataGrid project uses LCFG). The user can freely choose where to install the software (the rpm files are all relocatable); the restriction of having to be the superuser to install rpm files can be easily avoided by using a different RPM database, when root access is unavailable, or not advisable.
The rpm distribution of the CMS software has been already used successfully in a number of sites. The only issues were due to the use of obsolete, or buggy versions of RPM. The CMS rpm files could be used also on machines where part of the software was installed in a different way (e.g. from tarballs), but this required to disable the dependency checking.
In conclusion, the rpm distribution of the CMS software is mature enough to satisfy the requirements from the CMS collaboration and from the DataGrid project.